

ISSUES OF STRUCTURED VS. OBJECT-ORIENTED METHODOLOGY OF SYSTEMS ANALYSIS AND DESIGN

Mohammad A. Rob, University of Houston-Clear Lake, rob@cl.uh.edu

ABSTRACT

In recent years, there has been a surge of interest in object-oriented (OO) methodology in the analysis and design of a system. However, there is a dilemma as to how best fit the OO topics with the existing coherent discussion of structured approach of developing a system. This paper addresses some issues related to the two design paradigms.

Keywords: systems analysis and design, object-oriented design, waterfall methodology, systems development life cycle, SDLC.

INTRODUCTION

Systems Development Life Cycle (SDLC) is a methodology commonly used to develop an information system. Almost all texts used for the Systems Analysis and Design (SAD) course typically discuss and elaborate a structured approach called Waterfall model (4, 9, 13). These texts also organize their chapters sequentially according to the activities and phases (planning-analysis-design-implementation) performed in the Waterfall model. In recent years, there has been a rush of including object-oriented (OO) analysis and design in the SAD text. Some texts adapt object-oriented topics in various chapters as they fit best, while others include at the end of the text as an introduction to a new methodology. There is a dilemma as to how best fit the object-oriented topics with the existing coherent discussion of structured approach. The question is whether it is necessary or important to include the object-oriented design in the SAD texts that address structured approach or they should be addressed separately. This paper tries to find some answers to this question by addressing various issues related to the two design paradigms.

RESEARCH BACKGROUND

In a recent study, graduate students in a SAD course were asked to write a paper on a topic related to the systems analysis and design but that are not discussed in detail in the course. Table-1 presents a list of topics and the number of students who wrote the topics in three different semesters (Fall, 2002 - Fall, 2003). As seen, the majority of the students wrote on two topics – Joint Application Design (JAD) and Object-Oriented Design. Note, the three topics: Rational Unified Process, UML, and Use Cases, are all part of object-oriented design. The author believes that most serious students wrote on the latter topic. Thus object-oriented design methodology cannot be ignored in teaching Systems Analysis and Design.

TRADITIONAL APPROACH VERSUS OBJECT-ORIENTED APPROACH

The systems development life cycle (SDLC) can be defined as a process of understanding how an information system can support business needs or requirements of an organization,

modeling the business processes, designing the systems components, and building the system. A systems development project thus goes through a sequence of four fundamental phases: planning, analysis, design, and implementation. Each of these phases also consists of a series of steps or activities that rely on some techniques to produce required deliverables. Even though all projects cycle through some common phases or activities, but how they are approached by the systems development group can be different – the project team might move through the phases and steps logically, consecutively, incrementally, or iteratively (5).

TABLE-1: RESEARCH TOPICS AND THE NUMBER OF PAPERS IN EACH TOPIC

Topics	Number
Business Process Reengineering	7
Capability Maturity Model (CMM)	2
Change Management	2
COCOMO Model	4
Concurrent Software Development	1
Critical Chain Project Management	1
IDEFO	1
Joint Application Design	19
Motivation in Training	1
Outsourcing	1
Project Management	5
Prototyping	1
Rapid Application Development	3
Rational Unified Process	3
Root Cause Analysis	5
Scope Creep	5
Software Development Processes	1
Software Maintenance	1
Software Testing	2
Six Sigma	1
Spiral Model	1
Systems Implementation	1
UML: Object Oriented Approach	17
Unit Testing	1
Use Cases	1
Xtreme Programming	1

The most common approach to SDLC is a structured methodology that is based on waterfall model. It adopts a formal step-by-step approach to the SDLC phases and activities – the activities of one phase must be completed before moving to the next phase. See Figure 1. At the completion of each activity or phase, a document is produced that must be approved by the stakeholders before moving to the next activity or phase. The center of the structured approach is the process model that depicts the business processes of a system, and the tool used to present this model is the data-flow diagram (DFD). The DFDs and their associated data dictionary contain information about the systems components that need be designed and ultimately built. For example, data stores contribute to the design of files and databases, input data-flows contribute to the design of data-entry forms, and output data-flows contribute to the design of reports and statements, and the processes and other data-flows are used to design program modules. The structured approach has been around for many years and almost all SAD texts

describe this model in detail – the sections and chapters of these texts logically follow the phases and activities of SDLC.

Another approach to SDLC that is widely discussed in recent years is the object-oriented (OO) methodology. It is originated from the software engineering professionals who deal with systems development in various domains such as aerospace, business, and process control (7). Failure of a business system may cost money but that of mission-critical systems such as used in the airplanes and space shuttles might cost life. Business system is only one domain that is typically addressed in the SAD texts.

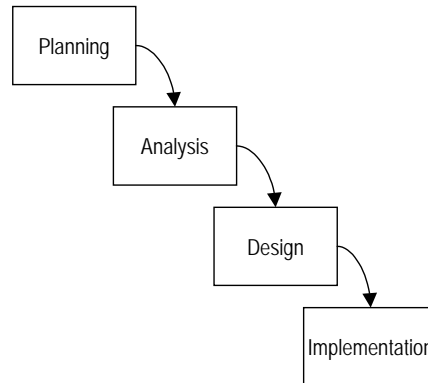


Figure 1: Structured approach to SDLC (Waterfall Model)

The object-oriented (OO) approach follows an iterative and incremental approach to systems development. The systems development life cycle is viewed as consisting of several increments or phases: inception, elaboration, construction, and transition (5). In each increment or phase, the developers move through the activities of gathering requirements, analyzing the requirements, designing the system, implementing the design, and testing the system. See Figure 2. Thus the phases of the traditional systems development approach do not match with those of the OO life cycle; but in each increment, all phases of the traditional life cycle (requirements, analysis, design, implementation, testing) are visited iteratively until the developers are satisfied. However, there are times when one activity predominates over the other four – causing the systems development effort to move from the inception to elaboration, then elaboration to construction, and from construction to transition.

The object-oriented approach uses a set of diagramming techniques known as the Unified Modeling Language or UML (1). It focuses on the three architectural views of a system: functional, static, and dynamic. The functional view describes the external behavior of the system from the perspective of the user. Use cases and use-case diagrams are used to depict the functional view. The static view is described in terms of attributes, methods, classes, relationships, and messages. Class-responsibility-collaboration (CRC) cards, class diagrams, and object diagrams are used to portray the static view. The dynamic view is represented by sequence diagrams, collaboration diagrams, and statecharts. All diagrams are refined iteratively until the requirements of the information system is fully understood. Finally, the information system is developed through a combination of traditional relational database and object-oriented programming – rather than true object-oriented methodology for both programming and database.

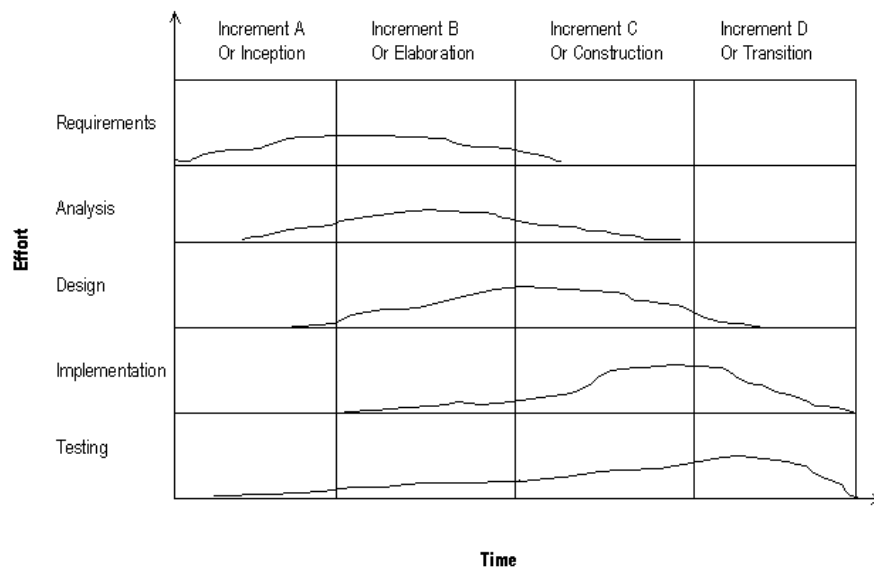


Figure 2: Iterative and Incremental Model of SDLC

DILEMMA BETWEEN THE TWO MODELS

There is a common misunderstanding that the object-oriented approach is the latest methodology of systems analysis and design. According to a recent survey, IT managers revealed that 39% of organizations have adopted OO methodology in some form; nevertheless, only 5% of IT projects are developed in OO methodologies (11). The Unified Processes are intended for use in developing large and complex information systems – handling many intricacies of the Unified modeling language might be far more complex than the development of the most information systems in business (8).

The steps of developing a system (planning-analysis-design-implementation) do not change but how they are performed can be different. The structured approach tries to understand a problem using a model termed Data-Flow Diagram (DFD), while the object-oriented approach uses Use Cases. In the structured approach, design of all system components (input, output, program modules, files and database) are evolved from the DFD, while in the UML approach, there are many models to deal with and there are no clear-cut steps such that the design of the system components evolve logically from a single model.

While the structured approach provides clear-cut steps (planning-analysis-design-implementation) from the beginning to the end of the systems development life cycle, the object-oriented approach does not – there is no clear-cut step in moving from analysis to design. The iterative and incremental nature of object-oriented approach refers to continuous testing and refinement of the system throughout the project lifecycle. The same UML diagramming techniques are used throughout all phases of the SDLC, although some diagrams are more important in some phases than the others. As the developers move through the SDLC, the diagrams gradually become more detailed, blurring the separation between the analysis and design phases. Understanding the continuous analysis of a problem through Use Cases is fine,

but when it comes to design, it becomes very complex – many models to focus on (9). Moving from design to implementation further complicates the matter. Why should we talk about the class model at the beginning of the analysis phase and then discuss about the relational database in the implementation phase – why not implement the system in an object-oriented database? Business systems are not developed in true object-oriented language such as PL/1 or smalltalk and object-oriented database; however, but most business systems are developed using Visual Basic (or C# or ASP.NET) programming language and relational database.

According to Dennis et. al. (5), UML is nothing more than a notation – it does not dictate any specific approach to developing information systems. Even though it is unlikely, it is possible to develop an information system using a traditional approach, such as structured systems development or information engineering, and to document the analysis and design using the UML. This reflects the intricacies of using true OO methodology in developing an information system.

DILEMMA BETWEEN THE SAD AND SOFTWARE ENGINEERING TEXTS

Software Engineering (or *Software Design*) is a text typically used in an introductory course in the Software Engineering program (2, 7, 12). In subsequent courses, the topics covered in the course are elaborated to have courses and hence texts such as requirement engineering, software engineering processes, and verification and validation. This defines the vigorousness necessary for the software engineering processes in mission-critical systems. Typically, there is only one Systems Analysis and Design course taught in an MIS program and recommended by MIS curricular models (3, 6). Only the *Software Engineering* text can be compared with the Systems Analysis and Design text.

UML defines a large set of diagramming techniques, but most object-oriented SAD texts focus on smaller set of the most commonly used techniques (5, 10). The major motivation behind the development of Unified Process was to present a methodology that could be used to develop large-scale information systems, typically, 500,000 lines of code or more (8).

Incorporating OO approach within the traditional approach in the same text complicates the subject matter of SDLC and confuses the reader. The phases of the traditional approach do not match with those of the OO approach. If we are to teach object-oriented approach, we should have a separate text for it and we should not try to compare the activities of the OO approach with the activities or phases of the structured approach.

CONCLUSION

In the era of object-oriented programming, object-oriented modeling is desirable, but mixing up structured methodology and OO methodology is confusing. What we need is a clear definition of models and steps as an analyst moves from the activities of analysis to design and from design to implementation.

REFERENCES

1. Booch, G, Jacobson, I. and Rumbaugh, J. (1999), *The Unified Modeling Language User Guide*, Addison-Wesley, Reading, MA.
2. Budgen, D, (2003), *Software Design, Second Edition*, Addison Wesley, New York.
3. Davis, G.B., Feinstein, D., Gorgone, J.T., Longenecker, Jr., H. E., & Valacich, J.S. (2002), IS 2002: An Update of the Information Systems Model Curriculum, *Proceedings of the Sixteenth Annual Conference of the International Academy for Information Management*, New Orleans, LA, pp. 76-82.
4. Dennis, A. and Wixom, B. H. (2000), *Systems Analysis and Design*, John Wiley & sons, New York.
5. Dennis, A., Wixom, B. H. and Tegarden, D. (2002), *Systems Analysis and Design: An Object-Oriented Approach*, John Wiley & sons, New York.
6. Longenecker Jr., H. E., Feinstein, D. L., Haigood, B., and Landry, J. P. (1999), On Updating the IS.97 Model Curriculum for Undergraduate Programs of Information Systems, *Journal of Information Systems Education*, (10:2), pp. 5-7.
7. Pressman, R (2000), *Software Engineering: A Practitioner's Approach*, Fifth Edition, McGraw Hill, New York.
8. Schach, S. R. (2004), *Introduction to Object-Oriented Analysis and Design with UML and the Unified Process*, Irwin- McGraw Hill, New York.
9. Satzinger, J. W. and Jackson, R. B. (2003), Making the Transition from OO Analysis to OO Design with the Unified Process, *Communications of the Association for Information Systems*, Volume 12, pp. 659-683.
10. Satzinger, J. W., Jackson, R. B., and Burd, S. D. (2002), *Systems Analysis and Design in a Changing World, Second Edition*, Course Technology, Boston, Massachusetts.
11. Sircar, S., Nerur, S.P., and Mahapatra, R. (2001), Revolution or Evolution? A Comparison of Object-Oriented and Structured Systems Development Methods, *MIS Quarterly*, Vol. 25, No. 4, pp. 457-471.
12. Sommerville, I. (2000), *Software Engineering*, 6th Edition, Addison Wesley, New York.
13. Whitten, j. L., Bentley, L. D. and Dittman, K. C. (2003), *Systems Analysis and Design Methods, Sixth Edition*, McGraw Hill, New York.